Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

Please amend claims 1, 2, 11, 15, 18, 19, 20, 34, and 36 as follows:

1. (currently amended): A method of indirect very long instruction word (VLIW) instruction memory (VIM) allocation comprising the steps of:

identifying a plurality of VLIW instructions in an input source program;

determining a lifetime of each of said plurality of VLIW instructions; and

allocating at least some of the plurality of VLIW instructions to VIM based on the lifetime ~~of each~~ of said plurality of VLIW instructions.

2. (currently amended): The method of claim 1 wherein the step of determining the lifetime of each of said plurality of VLIW instructions further comprises the steps of:

determining a control flow graph for the input source program containing said plurality of VLIW instructions;

determining a VLIW flow graph for said plurality of VLIW instructions; and

determining VLIW interference a graph.

3. (original): The method of claim 2 wherein the step of determining the VLIW flow graph further comprises the step of:

solving VLIW flow equations.

4. (original): The method of claim 2 wherein the control flow graph includes:

a plurality of nodes which correspond to basic blocks of the VLIW instructions; and

a plurality of edges, wherein each edge corresponds to a jump or a call from a given basic block to another basic block.

5. (original): The method of claim 4 wherein the flow control graph at each of said plurality of nodes includes:

at least one VLIW instruction defined by the node; and

at least one VLIW instruction used by the node.

6. (original): The method of claim 5 further comprising the step of:

determining live-in sets and live-out sets for each of said plurality of nodes.

7. (original): The method of claim 6 wherein the VLIW flow graph comprises the control flow graph and the live-in sets and live-out sets for each of said plurality of nodes.

8. (original): The method of claim 7 wherein the step of allocating VIM further includes the step of:

determining an interference graph in which every node of the interference graph corresponds to one of said plurality of VLIW instructions.

9. (original): The method of claim 8 wherein the VIM comprises a plurality of VIM lines, and the step of determining an interference graph further comprises the steps:

inserting an undirected edge into the interference graph between two VLIW nodes if the two VLIW instructions belong to a live-out set of the same node of the VLIW flow graph; and

coloring the interference graph nodes such that adjacent interference nodes are colored in different colors and each color corresponds to a different VIM line.

10. (original): The method of claim 1 wherein the lifetime of a VLIW instruction is a time interval extending from when said VLIW is defined by a load VLIW instruction to when said VLIW is last executed by an execute VLIW instruction.

11. (currently amended): The method of claim 1 further comprising the step of:

shortening the life of a particular VLIW by placing an initialization load VLIW (LV) statement adjacently prior to the use of its corresponding execute VLIW (XV) statement.

12. (original): The method of claim 1 further comprising the step of:

merging two non-overlapping VLIWs to share a common VIM line only when colorability of a resulting VLIW interference graph does not worsen as a result of said merging.

13. (original): The method of claim 1 further comprising the step of:

utilizing a coalescing heuristic to reduce VIM requirements of a program.

14. (original): The method of claim 13 wherein said step of utilizing a coalescing heuristic results in a coalesced VIM address holding two or more of said plurality of VLIW instructions.

15. (currently amended): A method of optimizing the execution time of a user program by reducing redundant loads of very long instruction word (VLIW) instruction memory (VIM) comprising the steps of:

selecting a load ~~VIM~~VLIW (LV) instruction in a current node; and

placing the LV instruction in a new node which is closer to a program start node if an execution frequency of the new node is lower than an execution frequency of the current node, and if a maximum number of VIM lines is not exceeded.

16. (original): A method to statically determine liveness of indirect very long instruction word (VLIW) instructions comprising the steps of:

determining a control flow graph which includes nodes representing basic program blocks, and edges connecting the nodes which represent jumps and calls from one block to another block; and

determining a VLIW flow graph by solving VLIW flow equations.

17. (original): The method of claim 16 wherein the VLIW flow equations comprise:

$I_n = U_n \cup (O_n - D_n)$; and

$O_n = \cup_{s \text{ in succ}(n)} I_s$;

where "n" is a given node, I_n is a set of live-in VLIWs at node "n", O_n is a set of live-out VLIWs at node "n", U_n is a set of VLIWs that are used in "n", D_n is a set of VLIWs that are defined in "n", the live-out VLIWs of node "n" are all the VLIWs that belong to live-in sets of successor nodes of "n", and the notation $\cup_{s \text{ in succ}(n)} I_s$ denotes the union of all sets I_s where s is a successor node to node n.

18. (currently amended): A method to statically determine interference of indirect very long instruction word (VLIW) instructions comprising the steps of:

determining an interference graph comprising VLIW nodes in which every VLIW node of the interference graph corresponds to one VLIW instruction[.];

inserting an undirected edge into the interference graph between two VLIW nodes if the two VLIW instructions belong to a live-out set of the same node of the VLIW flow graph; and

coloring the VLIW graph nodes such that adjacent VLIW nodes are colored in different colors and each color corresponds to a different VIM line.

19. (currently amended): An apparatus for allocating indirect very long instruction word (VLIW) instruction memory (VIM) comprising:

means for identifying a plurality of VLIW instructions in an input source program;

means for determining a lifetime of each of said plurality of VLIW instructions; and

means for allocating at least some of the plurality of VLIW instructions to VIM based on the lifetime ~~of each~~ of said plurality of VLIW instructions.

20. (currently amended): The apparatus of claim 19 wherein the means for determining the lifetime of each of said plurality of VLIW instructions further comprises:

means for determining a control flow graph for the input source program containing said plurality of VLIW instructions;

means for determining a VLIW flow graph for said plurality of VLIW instructions; and

means for determining a VLIW interference graph.

21. (original): The apparatus of claim 20 wherein the means for determining the VLIW flow graph further comprises:

means for solving VLIW flow equations.

22. (original): The apparatus of claim 20 wherein the control flow graph includes:

a plurality of nodes which correspond to basic blocks of the VLIW instructions; and

a plurality of edges, wherein each edge corresponds to a jump or a call from a given basic block to another basic block.

23. (original): The apparatus of claim 22 wherein the flow control graph at each of said plurality of nodes includes:

at least one VLIW instruction defined by the node; and

at least one VLIW instruction used by the node.

24. (original): The apparatus of claim 23 further comprising:

means for determining live-in sets and live-out sets for each of said plurality of nodes.

25. (original): The apparatus of claim 24 wherein the VLIW flow graph comprises the control flow graph and the live-in sets and live-out sets for each of said plurality of nodes.

26. (original): The apparatus of claim 25 wherein the means for allocating VIM further includes:

means for determining an interference graph in which every node of the interference graph corresponds to one of said plurality of VLIW instructions.

27. (original): The apparatus of claim 26 wherein the VIM comprises a plurality of VIM lines, and the means for determining an interference graph further comprises:

means for inserting an undirected edge into the interference graph between two VLIW nodes if the two VLIW instructions belong to a live-out set of the same node of the VLIW flow graph; and

means for coloring the interference graph nodes such that adjacent interference nodes are colored in different colors and each color corresponds to a different VIM line.

28. (original): The apparatus of claim 19 wherein the lifetime of a VLIW instruction is a time interval extending from when said VLIW is defined by a load VLIW instruction to when said VLIW is last executed by an execute VLIW instruction.

29. (original): The apparatus of claim 19 further comprising:

means for merging two non-overlapping VLIWs to share a common VIM line only when colorability of a resulting VLIW interference graph does not worsen as a result of said merging.

30. (original): The apparatus of claim 19 further comprising:

means for utilizing a coalescing heuristic to reduce VIM requirements of a program.

31. (original): The apparatus of claim 30 wherein said means for utilizing a coalescing heuristic produces a coalesced VIM address holding two or more of said plurality of VLIW instructions.

32. The apparatus of claim 19 further comprising:

means for shortening the life of a particular VLIW by placing an initialization LV statement adjacently prior to the use of its corresponding XV statement.

33. (original): An apparatus for optimizing the execution time of a user program by reducing redundant loads of very long instruction word (VLIW) instruction memory (VIM) comprising:

means for selecting a load VIM (LV) instruction in a current node; and

means for placing the LV instruction in a new node which is closer to a program start node if an execution frequency of the new node is lower than an execution frequency of the current node, and if a maximum number of VIM lines is not exceeded.

34. (currently amended): An apparatus for statically determining liveness of indirect very long instruction word (VLIW) instructions comprising:

means for determining a control flow graph which includes nodes representing basic program blocks containing VLIW instructions, and edges connecting the nodes which represent jumps and calls from one block to another block; and

means for determining a VLIW flow graph by solving VLIW flow equations.

35. (original): The apparatus of claim 34 wherein the VLIW flow equations comprise:

$I_n = U_n \cup (O_n - D_n)$; and

$O_n = \cup_{s \text{ in succ}(n)} I_s$;

where "n" is a given node, I_n is a set of live-in VLIWs at node "n", O_n is a set of live-out VLIWs at node "n", U_n is a set of VLIWs that are used in "n", D_n is a set of VLIWs that are defined in "n", the live-out VLIWs of node "n" are all the VLIWs that belong to live-in sets of successor nodes of "n", and the notation $\cup_{s \text{ in succ}(n)} I_s$ denotes the union of all sets I_s where s is a successor node to node n.

36. (currently amended): An apparatus statically determining interference of indirect very long instruction word (VLIW) instructions comprising:

means for determining an interference graph comprising VLIW nodes in which every VLIW node of the interference graph corresponds to one VLIW instruction[.];

means for inserting an undirected edge into the interference graph between two VLIW nodes if the two VLIW instructions belong to a live-out set of the same node of the VLIW flow graph; and

means for coloring the VLIW graph nodes such that adjacent VLIW nodes are colored in different colors and each color corresponds to a different VIM line.